UNIX - THE VI EDITOR TUTORIAL

There are many ways to edit files in Unix and for me one of the best ways is using screen-oriented text editor **vi**. This editor enable you to edit lines in context with other lines in the file.

Now a days you would find an improved version of vi editor which is called **VIM**. Here VIM stands for **V**i **IM**proved.

The vi is generally considered the de facto standard in Unix editors because:

- It's usually available on all the flavors of Unix system.
- Its implementations are very similar across the board.
- It requires very few resources.
- It is more user friendly than any other editors like ed or ex.

You can use **vi** editor to edit an existing file or to create a new file from scratch. You can also use this editor to just read a text file.

Starting the vi Editor:

There are following way you can start using vi editor:

Command	Description
vi filename	Creates a new file if it already does not exist, otherwise opens existing file.
vi -R filename	Opens an existing file in read only mode.
view filename	Opens an existing file in read only mode.

Following is the example to create a new file **testfile** if it already does not exist in the current working directory:

```
$vi testfile
```

As a result you would see a screen something like as follows:

```
|
~
~
~
~
~
~
~
~
~
~
~
"testfile" [New File]
```

You will notice a tilde (~) on each line following the cursor. A tilde represents an unused line. If a line does not begin

with a tilde and appears to be blank, there is a space, tab, newline, or some other nonviewable character present.

So now you have opened one file to start with. Before proceeding further let us understanding few minor but important concepts explained below.

Operation Modes:

While working with vi editor you would come across following two modes:

1. **Command mode:** This mode enables you to perform administrative tasks such as saving files, executing commands, moving the cursor, cutting (yanking) and pasting lines or words, and finding and replacing. In this mode, whatever you type is interpreted as a command.

2. **Insert mode:** This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and finally it is put in the file .

The vi always starts in command mode. To enter text, you must be in insert mode. To come in insert mode you simply type **i**. To get out of insert mode, press the **Esc** key, which will put you back into command mode.

Hint: If you are not sure which mode you are in, press the Esc key twice, and then you'll be in command mode. You open a file using vi editor and start type some characters and then come in command mode to understand the difference.

Getting Out of vi:

The command to quit out of vi is :q. Once in command mode, type colon, and 'q', followed by return. If your file has been modified in any way, the editor will warn you of this, and not let you quit. To ignore this message, the command to quit out of vi without saving is **:q!**. This lets you exit vi without saving any of the changes.

The command to save the contents of the editor is **:w**. You can combine the above command with the quit command, or :wq and return.

The easiest way to save your changes and exit out of vi is the **ZZ** command. When you are in command mode, type ZZ and it will do the equivalent of :wq.

You can specify a different file name to save to by specifying the name after the :w. For example, if you wanted to save the file you were working as another filename called filename2, you would type **:w filename2** and return. Try it once.

Moving within a File:

To move around within a file without affecting your text, you must be in command mode (press Esc twice). Here are some of the commands you can use to move around one character at a time:

Command	Description
k	Moves the cursor up one line.
j	Moves the cursor down one line.
h	Moves the cursor to the left one character position.
l	Moves the cursor to the right one character position.

There are following two important points to be noted:

- The vi is case-sensitive, so you need to pay special attention to capitalization when using commands.
- Most commands in vi can be prefaced by the number of times you want the action to occur. For example, 2j moves cursor two lines down the cursor location.

There are many other ways to move within a file in vi. Remember that you must be in command mode (press Esc twice). Here are some more commands you can use to move around the file:

Command	Description
0 or |	Positions cursor at beginning of line.
$	Positions cursor at end of line.
w	Positions cursor to the next word.
b	Positions cursor to previous word.
(	Positions cursor to beginning of current sentence.
)	Positions cursor to beginning of next sentence.
E	Move to the end of Blank delimited word
{	Move a paragraph back
}	Move a paragraph forward
[[	Move a section back
]]	Move a section forward
n|	Moves to the column n in the current line
1G	Move to the first line of the file
G	Move to the last line of the file
nG	Move to nth line of the file
:n	Move to nth line of the file
fc	Move forward to c
Fc	Move back to c
H	Move to top of screen
nH	Moves to nth line from the top of the screen
M	Move to middle of screen
L	Move to botton of screen
nL	Moves to nth line from the bottom of the screen

:x	Colon followed by a number would position the cursor on line number represented by **x**

Control Commands:

There are following useful command which you can use along with Control Key:

Command	Description
CTRL+d	Move forward 1/2 screen
CTRL+d	Move forward 1/2 screen
CTRL+f	Move forward one full screen
CTRL+u	Move backward 1/2 screen
CTRL+b	Move backward one full screen
CTRL+e	Moves screen up one line
CTRL+y	Moves screen down one line
CTRL+u	Moves screen up 1/2 page
CTRL+d	Moves screen down 1/2 page
CTRL+b	Moves screen up one page
CTRL+f	Moves screen down one page
CTRL+I	Redraws screen

Editing Files:

To edit the file, you need to be in the insert mode. There are many ways to enter insert mode from the command mode:

Command	Description
i	Inserts text before current cursor location.
I	Inserts text at beginning of current line.
a	Inserts text after current cursor location.
A	Inserts text at end of current line.
o	Creates a new line for text entry below cursor location.
O	Creates a new line for text entry above cursor location.

Deleting Characters:

Here is the list of important commands which can be used to delete characters and lines in an opened file:

Command	Description
x	Deletes the character under the cursor location.
X	Deletes the character before the cursor location.
dw	Deletes from the current cursor location to the next word.
d^	Deletes from current cursor position to the beginning of the line.
d$	Deletes from current cursor position to the end of the line.
D	Deletes from the cursor position to the end of the current line.
dd	Deletes the line the cursor is on.

As mentioned above, most commands in vi can be prefaced by the number of times you want the action to occur. For example, **2x** deletes two character under the cursor location and 2dd deletes two lines the cursor is on.

I would highly recommend to exercise all the above commands properly before proceeding further.

Change Commands:

You also have the capability to change characters, words, or lines in vi without deleting them. Here are the relevant commands:

Command	Description
cc	Removes contents of the line, leaving you in insert mode.
cw	Changes the word the cursor is on from the cursor to the lowercase w end of the word.
r	Replaces the character under the cursor. vi returns to command mode after the replacement is entered.
R	Overwrites multiple characters beginning with the character currently under the cursor. You must use **Esc** to stop the overwriting.
s	Replaces the current character with the character you type. Afterward, you are left in insert mode.
S	Deletes the line the cursor is on and replaces with new text. After the new text is entered, vi remains in insert mode.

Copy and Past Commands:

You can copy lines or words from one place and then you can past them at another place using following commands:

Command	Description
yy	Copies the current line.
yw	Copies the current word from the character the lowercase w cursor is on until the end of the word.
p	Puts the copied text after the cursor.
P	Puts the yanked text before the cursor.

Advanced Commands:

There are some advanced commands that simplify day-to-day editing and allow for more efficient use of vi:

Command	Description
J	Join the current line with the next one. A count joins that many lines.
<<	Shifts the current line to the left by one shift width.
>>	Shifts the current line to the right by one shift width.
~	Switch the case of the character under the cursor.
^G	Press CNTRL and G keys at the same time to show the current filename and the status.
U	Restore the current line to the state it was in before the cursor entered the line.
u	Undo the last change to the file. Typing 'u' again will re-do the change.
J	Join the current line with the next one. A count joins that many lines.
:f	Displays current position in the file in % and file name, total number of file.
:f filename	Renames current file to filename.
:w filename	Write to file filename.
:e filename	Opens another file with filename.
:cd dirname	Changes current working directory to dirname.
:e #	Use to toggle between two opened files.
:n	In case you open multiple files using vi, use :n to go to next file in the series.
:p	In case you open multiple files using vi, use :p to go to previous file in the series.
:N	In case you open multiple files using vi, use :N to go to previous file in the series.
:r file	Reads file and inserts it after current line

:nr file	Reads file and inserts it after line n.

Word and Character Searching:

The vi editor has two kinds of searches: string and character. For a string search, the / and ? commands are used. When you start these commands, the command just typed will be shown on the bottom line, where you type the particular string to look for.

These two commands differ only in the direction where the search takes place:

- The / command searches forwards (downwards) in the file.
- The ? command searches backwards (upwards) in the file.

The n and N commands repeat the previous search command in the same or opposite direction, respectively. Some characters have special meanings while using in search command and preceded by a backslash (\) to be included as part of the search expression.

Character	Description
^	Search at the beginning of the line. (Use at the beginning of a search expression.)
.	Matches a single character.
*	Matches zero or more of the previous character.
$	End of the line (Use at the end of the search expression.)
[	Starts a set of matching, or non-matching expressions.
<	Put in an expression escaped with the backslash to find the ending or beginning of a word.
>	See the '<' character description above.

The character search searches within one line to find a character entered after the command. The f and F commands search for a character on the current line only. f searches forwards and F searches backwards and the cursor moves to the position of the found character.

The t and T commands search for a character on the current line only, but for t, the cursor moves to the position before the character, and T searches the line backwards to the position after the character.

Set Commands:

You can change the look and feel of your vi screen using the following **:set** commands. To use these commands you have to come in command mode then type **:set** followed by any of the following options:

Command	Description
:set ic	Ignores case when searching
:set ai	Sets autoindent

:set noai	To unset autoindent.
:set nu	Displays lines with line numbers on the left side.
:set sw	Sets the width of a software tabstop. For example you would set a shift width of 4 with this command: **:set sw=4**
:set ws	If *wrapscan* is set, if the word is not found at the bottom of the file, it will try to search for it at the beginning.
:set wm	If this option has a value greater than zero, the editor will automatically "word wrap". For example, to set the wrap margin to two characters, you would type this: **:set wm=2**
:set ro	Changes file type to "read only"
:set term	Prints terminal type
:set bf	Discards control characters from input

Running Commands:

The vi has the capability to run commands from within the editor. To run a command, you only need to go into command mode and type **:!** command.

For example, if you want to check whether a file exists before you try to save your file to that filename, you can type **:! ls** and you will see the output of ls on the screen.

When you press any key (or the command's escape sequence), you are returned to your vi session.

Replacing Text:

The substitution command (**:s/**) enables you to quickly replace words or groups of words within your files. Here is the simple syntax:

```
:s/search/replace/g
```

The g stands for globally. The result of this command is that all occurrences on the cursor's line are changed.

IMPORTANT:

Here are the key points to your success with vi:

- You must be in command mode to use commands. (Press Esc twice at any time to ensure that you are in command mode.)
- You must be careful to use the proper case (capitalization) for all commands.
- You must be in insert mode to enter text.